Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **7**, Number **1**, February **2022**

Conditional Adversarial Domain Adaption based on Self-attention

Li-Quan Zhao

Key Laboratory of Modern Power System Simulation and Control & Renewable Energy Technology, Ministry of Education
Northeast Electric Power University
Jilin 132012, China
zhaoliquan@neepu.edu.cn

Yu-Peng Zhang*

Key Laboratory of Modern Power System Simulation and Control & Renewable Energy Technology, Ministry of Education
Northeast Electric Power University
Jilin 132012, China
Corresponding Author: 207205933@qq.com

Zi-Ming Teng

College of information and communication
Jilin University
Changchun, 130012, China
tengziming2002@163.com

Zi-Cong Jiang

Graduate school of Information Science and Electrical Engineering
Kyushu University
Fukuoka 819-0395, Japan
jiangzicong1234@gmail.com

Ying Cui

Zhuhai Power Supply Bureau
Guangdong Electric Power Corporation
Zhuhai 519000, China
cuiying794758706@126.com

Zhong-Feng Kan

Jilin Power Supply Bureau
Jilin Electric Power Corporation
Jilin 132012, China
kanzhongfeng@163.com

Received October 2021; revised November 2021

Abstract. *Transfer learning is a deep learning method. It transforms knowledge in the source domain and feeds it to the target domain to improve the performance of the target domain model. The conditional adversarial domain adaption method is one of a number of transfer learning methods. It uses Resnet as the feature extraction network; however, Resnet can only determine the relationship between local features, and so its feature extraction accuracy is limited. To overcome this problem, a self-attention layer has been designed in order to determine the relationship among all features. This layer is added between the fourth layer and fifth layer of Resnet. Besides, a cross-entropy function is introduced into loss function, and a sigmoid layer is designed and used as the out layer in conditional adversarial domain adaption. These produce gradient disappearance in the deep network, and influence the discriminator training effect. To solve that problem, a least squares function is used as to design the loss function. The new type of loss function does not require the sigmoid function. Using theoretical analysis, we explain how this new loss function can be used to solve the gradient disappearance problem, thus rendering the network structure more stable. We compare our proposed method with other datasets, including Joint Adaptation Networks, adversarial domain adaptation, conditional adversarial domain adaption, Resnet-50, domain-adversarial neural network on imageCLEF-DA, Office-Home, and Office-31. Our simulation results show that the proposed conditional adversarial domain adaption based on self-attention method has the fastest convergence speed and highest accuracy the among different datasets.*
Keywords: Deep Learning; Computer Vision; Transfer learning; Adversarial network

1. **Introduction.** Deep learning can accomplish different machine learning tasks by constructing a deep neural network. It has huge development potential in autonomous driving [1, 2], image caption [3], object detection [4] and solar irradiance prediction [5]. In deep learning problems, it assumes that the data distributions of the training set and test set are the same. However, in reality there are some differences between these data distributions, resulting in differences in model performance between the training set and test set. In order to solve this problem, transfer learning as a new learning framework has been proposed [6]. Domain adaptation is one such transfer learning method, which involves a labeled dataset (source domain) and a different yet related, unlabeled dataset (target domain). The source domain and target domain have the same feature spaces and category space, but different feature distributions [7].

The Domain adaptation method includes four classifications. The first classification is class criterion, which uses label information as a guide to transfer the knowledge from source domain to target domain [8–10]. The second classification is statistic criterion, which defines the target and source domain statistical distributions by using some standard alignment methods, such as MMD, CORAL, KL divergence [11–14]. The third classification is architecture criterion, which improves the methods ability of transferring features by adjusting the deep network structure [15, 16]. The fourth classification is adversary network module, a technique which adds generative adversarial networks (GANs) containing the discriminative network and generative network into the domain adaption network. The generative network continues to learn the data features of the target domain and source domain, thus preventing the discriminator from distinguishing the features of the two domains. This makes the generated features more representatives, and reduces error. Compared with the generation adversarial networks, the domain adaptation directly uses the data in the target domain as the generated samples to avoid generating samples. Therefore, the purpose of the generator has been changed. It is no longer used to generate samples but to extract feature.

Generative adversarial network (GAN) has developed rapidly in recent years. It has been successfully applied in many fields. Adversarial domain adaptation is one of its successful applications. It belongs to the fourth category of domain adaption classification. This is the one that we focus on. There are a number of methods that belong to this category. The domain-adversarial neural network (DANN) is perhaps the first in which generative adversarial networks have been incorporated into domain adaption [17]. It proposes the basic structure of domain adaptation including feature extractor, category predictor and domain discriminator. It adds a domain discriminate network after feature extraction in order to determine whether the data comes from the source domain or the target domain. In recent years, most adversarial domain adaptations are modeled according to this network architecture. The idea of using adversarial for domain adaptation is widely used. It is more like a framework for the domain adaption domain, where the researchers are constantly adding various networks with directional functions to accomplish specialized tasks. Adversarial discriminative domain adaptation (ADDA) comprises a common framework for domain adaptive methods using discriminators [18]. It consists of a discriminative model, untie weight sharing, and GAN loss function. Based on DANN, the ADDA broadens the scope of adversarial domain adaption network architecture, and is not limited to one structure. Researchers can construct different models in the adversarial domain adaption according to collocation methods such as whether to use Weight Share and which loss function is selected. Developing on the DANN method, Long, et al. proposed the conditional adversarial domain adaption (CDAN) method [19] by replacing the generative adversarial networks in DANN with conditional generative adversarial networks, and by adding features and labels in the network. The label information can be used to align features. However, with both DANN and CDAN methods the target domain label space is only a subset of the source domain label space; in an effort to solve this problem, Cao et al. [20] proposed the selection adversarial network (SAN) method, one which designs domain discriminators (the number of source domain labels). Each discriminator has a weight, which helps to prevent those categories that are not in the target domain from participating in the transfer. Similar methods include Wasserstein distance guided representation learning [21], cycle-consistent adversarial domain adaptation [22] and Unsupervised image-to-image translation networks [23].

The mainly contributions of our manuscript have two points: 1) the conditional adversarial domain adaptation method only considers feature relationships between the current pixel and pixels closer to the current pixel, which failures to capture the remote pixel information relationship. It affects classification accuracy of conditional adversarial domain adaptation method. To solve the problem, we design a self-attention layer and add it between the fourth layer and fifth layer of Resnet-50 network to determine the relationship among all features. Compared with original conditional adversarial domain adaptation method, it can effectively extract feature relationships between the current pixel and all other pixels, which improve the classification accuracy. 2) The conditional adversarial domain adaptation method uses cross-entropy function as loss function and sigmoid layer as the output layer and ignores the problem caused by distance between the decision boundary and the extracted features, which easily generates gradient disappearance in the deep network and influences the discriminator performance. To solve the problem, we use least squares function as the loss function to solve the gradient disappearance problem. It has the advantages of preventing gradient dissipation, increasing feature sharing and shortening the distance between extracted features and decision boundary. Compared with the original conditional adversarial domain adaptation method, our proposed method has better performance in accuracy and convergence speed.

2. **Conditional adversarial domain adaption.** In conditional adversarial domain adaptation [19], when the data features form a very complex structure, domain adaption based on an adversarial network cannot capture the multi-modal data structure. This results in negative transfer and a large margin of error. Ganin first introduced the idea of confrontation into domain adaption, and proposed the DANN method [17], which in turn consists of source classifier, feature extractor and domain discriminator. In proposing conditional adversarial domain adaptation, they pointed out that the DANN method only focuses on the overall distribution of data characteristics and ignores the correlation between the classes, at the expense of accuracy. With conditional adversarial domain adaptation, the features and classes could be adaptively adjusted together.

The loss function of conditional adversarial domain adaptation is as follows:

$$\min_G \varepsilon(G) - \lambda\varepsilon(D,G) \tag{1}$$

$$\min_D \varepsilon(D,G) \tag{2}$$

where G expresses the source classifier, D expresses the domain discriminator and λ is a hyper-parameter between the two objectives. The $\varepsilon(G)$ and $\varepsilon(D,G)$ are expressed as followings:

$$\varepsilon(G) = E_{(x_i^s,y_i^s)\sim D^s} L(G(x_i^s), y_i^s) \tag{3}$$

$$\varepsilon(D,G) = -E_{x_i^s\sim D^s} \log\left[D\left(f_i^s, g_i^s\right)\right] - E_{x_j^t\sim D^T} \log\left[1 - D\left(f_j^t, g_j^t\right)\right] \tag{4}$$

where $x_i^s \sim D^s$ is the data of source domain, $x_j^t \sim D^T$ is the data of target domain, $L(.,.)$ expresses cross-entropy loss function. $f = F(x)$ represents features, $g = G(x)$ represents labels generated from source classifier G. The function $f \otimes g$ represents the multilinear map. The conditional adversarial domain adaptation (CDAN) method merely introduces features and labels to the network via multilinear mapping. However, one disadvantage of the multilinear map is dimensional explosion. In deep learning, the dimensions of features and labels are always large, and gradient explosions will be unavoidable when features and labels are multiplied via tensors. To reduce computational complexity, CDAN method introduces the following conditioning strategy that is shown in the following:

$$T(h) = \begin{cases} T_{\otimes}(f,g) = f \otimes g & d_f \times d_g \leq 4096 \\ T_{\odot}(f,g) = \frac{1}{\sqrt{d}}(R_f f) \odot (R_g g) & otherwise \end{cases} \tag{5}$$

where d_f and d_g are the dimensions of feature vector f and label vector g, respectively. d is multilinear map of dimension. R_f and R_g are random matrices sampled from the features and labels, and $\odot$ represent element-wise produce.

Besides, the conditional adversarial domain adaptation method also introduce entropy to the network. The entropy can be expressed as followings:

$$\begin{aligned} H(g) &= -\sum_{c=1}^{C} g_c \log g_c \\ w(H(g)) &= 1 + \exp(-H(g)) \end{aligned} \tag{6}$$

where g_c is probability that an example is predicted to class c, and C is a classed number. The entropy condition finally added to the network is:

$$\omega(H(g)) = 1 + \exp(-H(g)) \tag{7}$$

Finally, the conditional adversarial domain adaptation loss function is defined:

$$\begin{aligned} \min_G E_{\left(x_i^s,y_i^s\right)\sim D_s} L\left(G\left(x_i^s\right), y_i^s\right) + \lambda E_{x_i^s\sim D^s} \omega\left(H\left(g_i^s\right)\right) \log\left[D\left(T\left(h_i^s\right)\right)\right] + \\ \lambda E_{x_j^t\sim D^t} \omega\left(H\left(g_j^t\right)\right) \log\left[D\left(T\left(h_j^t\right)\right)\right] \end{aligned} \tag{8}$$

$$\max_{D} E_{x_i^s \sim D^s} \omega\left(H\left(g_i^s\right)\right) \log\left[D\left(T\left(h_i^s\right)\right)\right] + E_{x_j^t \sim D^t} \omega\left(H\left(g_j^t\right)\right) \log\left[D\left(T\left(h_j^t\right)\right)\right] \tag{9}$$

The frameworks of conditional adversarial domain adaptation are shown in Figure 1 and Figure 2.

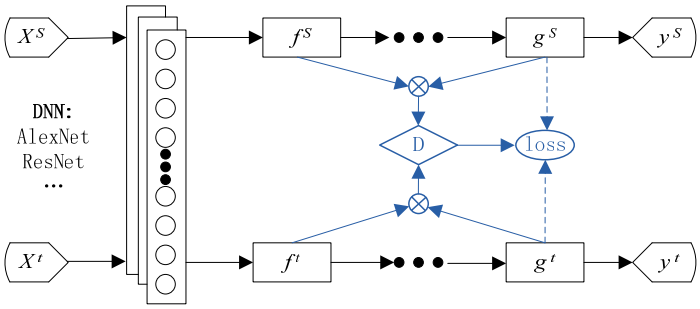

FIGURE 1. Network structure of CDAN in low-dimensional scenario

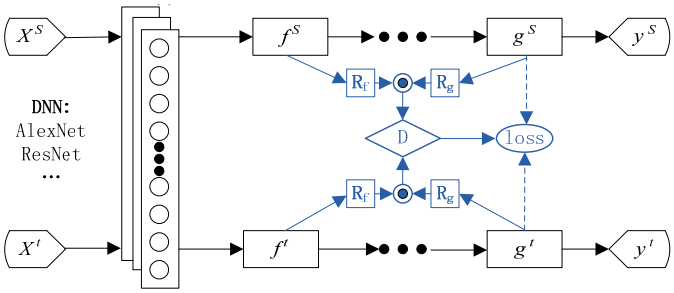

FIGURE 2. Network structure of CDAN high-dimensional scenario

3. **Our proposed method.**

3.1. **Design of self-attention layer.** Conditional adversarial domain adaptation employs Resnet as its feature extraction network. It uses only the relationship of local features to update extracted features in Resnet, an approach which can undermine feature extraction accuracy. To overcome this problem, we design a self-attention neural network that uses the relationship between all features to update extracted features, and feeds those into the Resent network via conditional adversarial domain adaptation. The operation of self-attention neural networks can be written as:

$$\gamma_{i,j} = \frac{f(x_i, x_j)}{\sum_{j=1}^{N} f(x_i, x_j)} \tag{10}$$

where x_i is the ith component of the feature matrix x that is the output of one Resnet network layer, x_j is the jth component of the feature matrix, and N is the number of all components of the feature matrix. As shown in Eq.(12), $\gamma_{i,j}$ is not only affected by x_i but also by x_j. The equation shows the correlation between feature components x_i and x_j.The Self-attention operation is different from that of the fully-connected network layer and convolutional network layer. The fully-connected network layer uses the learned weight to calculate the mapping between output and input. The convolutional layer is a weighted sum of eigenvalues in a local neighborhood.

The self-attention module can be seen as a special form of the non-local embedded Gaussian version, in that $f(x_i, x_j)$ can be expressed as:

$$f(x_i, x_j) = e^{\theta(x_i)^T \phi(x_j)} \tag{11}$$

where

$$\theta(x_i) = W_\theta \times x_i \tag{12}$$

$$\phi(x_j) = W_\varphi \times x_j \tag{13}$$

where W_θ and W_φ represent the convolution, whereby each convolution kernel size is 1×1 and the number of channels is one eight of input channel,$\theta(x_i)$, $\varphi(x_j)$ are two feature spaces obtained by the convolution operation in order to reduce complexity. This is realized by reducing the parameters and channels of each convolution kernel.

Therefore, based on Eq. (13) to Eq. (15), Eq. (11) is expressed as:

$$\gamma_{i,j} = \frac{\exp[\theta(x_i)^T \phi(x_j)]}{\sum_{j=1}^{N} \exp[\theta(x_i)^T \phi(x_j)]} = \frac{\exp[(Wx_i)^T (Wx_j)]}{\sum_{j=1}^{N} \exp[(Wx_i)^T (Wx_j)]} = soft\max(x_i^T W^T W x_j) \tag{14}$$

where $\gamma_{i,j}$ is a self-attention map. Based on Eq. (16), the output of our proposed attention layer is:

$$o_i = \sum_{j=1, j\neq i}^{\mathrm{N}} \gamma_{i,j} h(x_j) = \sum_{j=1, j\neq i}^{\mathrm{N}} soft\max(x_i^T W_\theta^T W_\phi x_j) h(x_j) \tag{15}$$

where $h(x_j) = W_h x_j$, W_h is also convolution that the size of convolution kernel and number of channel are the same with W_θ and W_φ.

Finally, it combines self-attention layer output o and input feature x to get the output feature:

$$y_i = W_o o_i + x_i \tag{16}$$

where W_0 is 1×1 convolution. The number of channels for W_0 is two times that of channels for W_h. y_i is the output feature of Resnet network. We add the proposed self-attention layer between the fourth layer and fifth layer of the Resnet network, which is shown in Figure 3. In Figure 3, the leftmost and rightmost networks are the fourth layer and fifth layer of the Resnet network, respectively. The Resnet network is used to extract feature in conditional adversarial domain adaptation method. The middle networks are our proposed networks in Figure 3. The x_i is the output feature of the fourth layer. The first convolution is used to reduce the number of channels in order to reduce compute complexity. The $\gamma_{i,j}$ obtained by using Eq. (16) is the output of the first convolution with sigmoid active function. The second convolution is used to increase the number of channels in order to restore the image channel number so that it can be better connected to the next layer. The input of the self-attention layer comprises the output features of the fourth layer of Resnet, and the output of self-attention layer comprises the input features of the fifth layer of Resnet.

3.2. **Proposed loss function.** In the loss function of Conditional adversarial domain adaptation, it uses cross-entropy function as nonlinear function, and a sigmoid layer as the last layer in the discriminator. With the cross-entropy loss function, with the increase of iteration number, the gradient of function will approach zero, by which point the gradient cannot be updated to the network, and the model will not perform well. Another problem is that some of the features obtained by cross entropy are correctly classified,

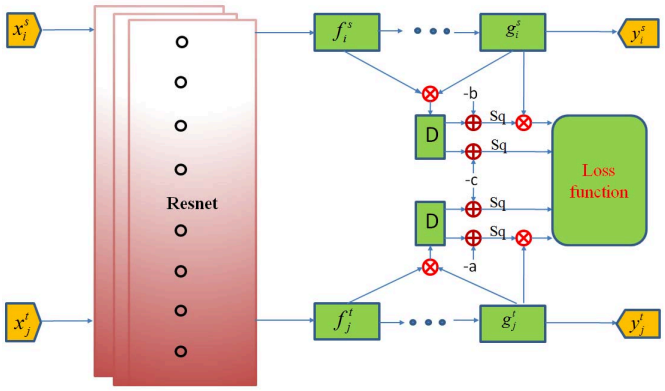

FIGURE 3. Self-attention layer

but they are far away from the decision boundary (which is a boundary that can correctly classify samples). The function of the decision boundary is to distinguish between the features of the source domain and those of the target domain; this simplifies the process of determining which domain they come from. However, this discriminator is affected by two problems: gradient disappearance, and collapse mode [24].

To solve those problems, we propose using the least squares function as the loss function in adversarial domain adaption, and deleting the sigmoid layer in the discriminator. The least squares-based loss function will shorten and improve the feature component classification process, albeit further away from the decision boundary, so that the whole feature component is closer to the boundary than before (i.e. after the features are processed, their distribution is more difficult to distinguish in terms of the domains from which they originate, thus the extracted feature quality is improved). The least squares loss function of our proposed method consists of three terms. The first one is expressed as follows:

$$\varepsilon(G) = \frac{1}{2} E_{(x_i^s, y_i^s) \sim D_s} D(G(x_i^s), y_i^s)^2 \tag{17}$$

Where x_i^s is the source data, y_i^s is its corresponding label, $G()$ is the source classifier, $D()$ is the discriminator, $\varepsilon(G)$ is the source domain loss function used to measure the source classifier performance. f_i^s is the feature representation of the source domain, and g_i^s is the classifier prediction obtained by source classifier $G()$.

The second term is expressed as follows:

$$\gamma(D) = -\frac{1}{2} E_{x_i^s \sim D_s} \omega\left(H\left(g_i^s\right)\right)\left(D\left(f_i^s, g_i^s\right) - b\right)^2 - \frac{1}{2} E_{x_j^t \sim D_t} \omega\left(H\left(g_j^t\right)\right)\left(D\left(f_j^t, g_j^t\right) - a\right)^2 \tag{18}$$

where f_j^t is the feature representation of the target domain obtained by feature extractor $\bar{G}$, and g_j^t is the classifier prediction obtained by target classifier $G()$; b is label of the target domain, and a is label of the source domain, $H()$ is standard entropy, and $\omega(x) = 1 + e^{-x}$ expresses Entropy-aware weight. The last term is expressed as follows:

$$\gamma\left(\bar{G}\right) = -\frac{1}{2} E_{\left(x_j^t \sim D_t\right)}\left(D\left(f_j^t, g_j^t\right) - c\right)^2 - \frac{1}{2} E_{\left(x_i^s \sim D_s\right)}\left(D\left(f_i^s, g_i^s\right) - c\right)^2 \tag{19}$$

where $\bar{G}$ is the feature extractor, c is the value that domain discriminator D believes to extract features from source domain; and $\varepsilon(D)$, $\gamma(D)$ and $\gamma\left(\bar{G}\right)$ are used to measure the classifier, discriminator and feature extractor performances, respectively.

Based on the above three terms, our proposed least squares loss function is expressed as follows:

$$\min_{G} \varepsilon(G) - \lambda[\gamma(D) + \gamma(\overline{G})] \tag{20}$$

$$\min_{\overline{G}} \min_{D} [\gamma(D) + \gamma(\overline{G})] \tag{21}$$

where λ is a hyper-parameter that is used to establish a tradeoff between domain adversary and source risk. The network of our proposed method is illustrated in the Figure 4.

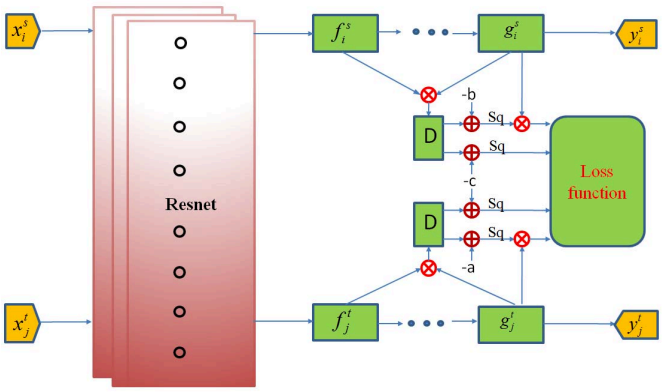

FIGURE 4. Module architecture of our proposed method

In the following, we will prove the reason why the least squares loss function can avoid the phenomenon of gradient disappearance from the theoretical level. In the conditional adversarial domain adaptation, optimization of the loss function in generative adversarial networks is based on Jensen-Shannon divergence [24]:

$$2JS(x_i^s \sim D_s || x_j^t \sim D_t) - 2\log 2 \tag{22}$$

when the distribution of $x_i^s \sim D_s$ is close to $x_j^t \sim D_t$, the Jensen-Shannon divergence is close to zero and the gradient will disappear. This can undermine network stability performance. By optimizing our proposed loss function, this problem can be eliminated.

Firstly, we take a derivative with respect to D in order to obtain the optimal discriminator D^*, which is expressed as follows:

$$D^*(x) = \frac{b(x_i^s \sim D_s) + a(x_j^t \sim D_t)}{(x_i^s \sim D_s) + (x_j^t \sim D_t)} \tag{23}$$

Secondly, we introduce Eq. (25) into Eq. (21) and make the following deductions:

$$\begin{aligned}
&2\gamma\left(\bar{G}\right) \\
&= -E_{\left(x_j^t \sim D_t\right)}\left(D\left(f_j^t, g_j^t\right) - c\right)^2 - E_{\left(x_i^s \sim D_s\right)}\left(D\left(f_i^s, g_i^s\right) - c\right)^2 \\
&= -E_{\left(x_j^t \sim D_t\right)}\left[\left(\frac{b\left(x_i^s \sim D_s\right) + a\left(x_j^t \sim D_t\right)}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)} - c\right)^2\right] - E_{\left(x_i^s \sim D_s\right)}\left[\left(\frac{b\left(x_i^s \sim D_s\right) + a\left(x_j^t \sim D_t\right)}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)} - c\right)^2\right] \\
&= -\int_{\chi}\left(x_j^t \sim D_t\right)\left(\frac{(b-c)\left(x_i^s \sim D_s\right) + (a-c)\left(x_j^t \sim D_t\right)}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)}\right)^2 dx - \int_{\chi}\left(x_i^s \sim D_s\right)\left(\frac{(b-c)\left(x_i^s \sim D_s\right) + (a-c)\left(x_j^t \sim D_t\right)}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)}\right)^2 dx \\
&= -\int_{\chi}\frac{\left((b-c)\left(x_i^s \sim D_s\right) + (a-c)\left(x_j^t \sim D_t\right)\right)^2}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)} dx \\
&= -\int_{\chi}\frac{(b-c)\left(\left[\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)\right] - (b-a)\left(x_j^t \sim D_t\right)\right)^2}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)} dx
\end{aligned} \tag{24}$$

Finally, we set $a = -1$, $b = 1$ and $c = 0$, and simplify Eq. (26) as follows:

$$\begin{aligned} 2\gamma\left(\bar{G}\right) Z &= -\int_{\chi} \frac{\left(2\left(x_j^t \sim D_t\right) - \left[\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)\right]\right)^2}{\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)} dx \\ &= \chi^2_{pearson}\left(\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right) || 2\left(x_j^t \sim D_t\right)\right) \end{aligned} \tag{25}$$

As shown in Eq. (27), the process of minimizing the target function involves minimizing the Pearson χ^2 divergence between $\left(x_i^s \sim D_s\right) + \left(x_j^t \sim D_t\right)$ and $2\left(x_j^t \sim D_t\right)$. The Pearson χ^2 has no gradient disappearance problem [24]; therefore, our proposed loss function should not be affected by the gradient disappearance problem.

4. **Experimental results and analysis.** We test the performances of our proposed method and other five methods that are Resnet-50 method [25], Deep Adaptation Network (DAN) method [10], Joint Adaptation Networks (JAN) method [13], Domain-Adversarial Neural Network (DANN) method [17], Conditional Adversarial Domain Adaption (CDAN) method [19] on three different datasets(Office-Home dataset, Office-31 dataset and ImageCLEF-DA dataset). Office-31 dataset is a benchmark dataset that is commonly used in domain adaptation. Office-31 consists of 31 classes and 4652 images. It consist of three domains that are DSLR(D),Webcam(W) and Amazon(A). These domains complete the following tasks: D to W task, W to D task, D to A task, A to D task, W to A task and A to W task. ImageCLEF-DA dataset is made up of 12 classes and three datasets: Pascal VOC 2012 (P),ILSVRC 2012 (I) and caltech-256 (C). Therefore, it has six tasks that are P to I task, P to C task, I to P task, I to C task, C to I task and C to P task. The The Office-Home dataset is a complex dataset that includes 65 classes and 15500 images. It consist of four datasets that are Product images (P), Clip Art (C), real-word images (R) and Artistic images (A). The four datasets are more different between each other than Office-Home dataset and Office-31 dataset. Therefore, It is large challenge for current methods. It has 12 tasks that are P to C task, P to R task, P to A task, C to R task, C to A task, R to A task, A to R task, A to C task, A to P task, R to C task, R to P task and C to P task. The basic network adopted for all tasks is Resnet-50, and all experiments are carried out using pytorch.

For our experiment, we adopt the same protocols that are used in conditional adversarial domain adaptation, and the pre-train network is finetuned using ImageNet. The source domain data have been labeled, and that of the target domain do not have been labeled. The classification accuracy results are generated in five random experiments. For the loss function, the transfer loss and classifier loss are of equal importance. We adopt importance-weighted cross-validation to select hyper-parameters. In our experiment the mini-SGD is identical to the conditional adversarial domain adaptation (batch-size = 32, momentum = 0.9,learning rate = 0.001, weight decay = 0.0005). Our operating system is Ubuntu and the GPU is NVIDIA GTX 2080Ti.The basic network used is ResNet-50, and the deep learning framework is PyTorch.

4.1. **Comparison of accuracy.** The accuracy results for our proposed method and those of other domain adaption methods(CDAN method, JAN method, DAN method, DANN method and Resnet-50 method) on the Office-31 dataset are shown in Table I. For A to W task, D to W task, A to D task, D to A task and W to A task, our proposed method still has the highest accuracy. For W to D task, the accuracy of our proposed method is 100%, and that of CDAN method is also 100%. They have the same accuracy. It also can be seen that the average accuracies are 89.0%, 86.6%, 76.1%, 82.2%, 84.3% and 80.4% for our proposed method, CDAN method, Resnet-50 method, DANN method, JAN method and

DAN method,respectively. This means that our proposed method has the best accuracy performance on the Office-31 dataset.

The accuracy results for our proposed method and other methods(CDAN method, JAN method, DAN method, DANN method and Resnet-50 method) on the imageCLEF-DA dataset are shown in Table 2. For all six tasks, our proposed method still has the highest accuracy and average accuracy, followed by CDAN method. Compared with CDAN method,the accuracies of our proposed method are 0.1%, 0.9%, 1.0%, 1.3%, 1.6% and 1.7% higher than that of CDAN for I to C task, P to C task, P to I task, C to P task, I to P task AND C to I task, respectively. The average accuracies of our proposed method and CDAN method are 88.2% and 87.1%, respectively. Therefore, the above results show that our proposed method has the best performance in accuracy on imageCLEF-DA dataset.

Table 1. Accuracy (%) on office-31 dataset.

Method	W to A	W to D	A to W	A to D	D to W	D to A	Avg
Proposed method	73.1	100.0	95.2	94.1	98.6	73.0	89.0
CDAN	68.0	100.0	93.1	89.8	98.2	70.1	86.6
Resnet-50	60.7	99.3	68.4	68.9	96.7	62.5	76.1
DANN	67.4	99.1	82.0	79.7	96.9	68.2	82.2
JAN	70.0	99.8	85.4	84.7	97.4	68.6	84.3
DAN	62.8	99.6	80.5	78.6	97.1	63.6	80.4

Table 2. Accuracy (%) on ImageCLEF-DA dataset.

Method	C to P	C to I	I to P	I to C	P to C	P to I	Avg
Proposed method	75.8	92.2	78.3	97.1	91.6	94.4	88.2
CDAN	74.5	90.5	76.7	97.0	90.6	93.5	87.1
Resnet-50	65.5	78.0	74.8	91.5	83.9	91.2	80.7
DANN	74.3	87.0	75.0	96.2	86.0	91.5	85.0
JAN	74.2	89.5	76.8	94.7	88.0	91.7	85.8
DAN	69.2	86.3	74.5	92.8	82.2	89.8	82.5

The accuracy results on the the Office-Home dataset for different methods(CDAN method, JAN method, DAN method, DANN method and Resnet-50 method) are shown in Table 3. For all tasks, our proposed method still has the highest accuracy and average accuracy, followed by CDAN method. Compared with CDAN method,the accuracies of our proposed method are 3.0%, 3.3%, 1.9%, 5.0%, 6.5%, 4%, 4.4% ,5.7%, 2.5%, 4.8%, 2.7% and 2.2% higher than that of CDAN for A to C task, A to P task, A to R task, C to A task, C to P task, C to R task, P to A task, P to C task, P to R task, R to A task, R to C task and R to P task, respectively. Besides, the average accuracies of our proposed method and CDAN method are 67.6% and 63.8%, respectively. the average accuracy of our proposed method is 3.8% higher than that of CDAN method. Therefore, the above results show that our proposed method has the best performance in accuracy on imageCLEF-DA dataset. On the Office-31 dataset and image-CLEF-DA dataset, the average accuracy of the proposed method is 89.0% and 88.2%, which are 2.4% and 1.1% higher than that of CDAN method, respectively. The reason for this is that the Office-Home dataset is the most complex of the three datasets: in the Office-Home dataset there

is a large difference between the three domains, and the categories have less in common, so there is room for improvement.

TABLE 3. Accuracy (%) on Office-Home dataset

	JAN	DAN	ResNet-50	DANN	CDAN	Proposed method
A to P	61.2	57.0	50.0	59.3	69.3	72.6
A to C	45.9	43.6	34.9	45.6	49.0	52.0
A to R	68.9	67.9	58.0	70.1	74.5	76.4
C to A	50.4	45.8	37.4	47.0	54.4	59.4
C to P	59.7	56.5	41.9	58.5	66.0	72.5
C to R	61.0	60.4	46.2	60.9	68.4	72.4
P to A	45.8	44.0	38.5	46.1	55.6	60.0
P to C	43.4	43.6	31.2	43.7	48.3	54.0
P to R	70.3	67.7	60.4	68.5	75.9	78.4
R to P	76.8	74.3	59.9	76.8	80.5	82.7
R to C	52.4	51.5	41.2	51.8	55.4	58.1
R to A	63.9	63.1	53.9	63.2	68.4	73.2
Avg	58.3	56.3	46.1	57.6	63.8	67.6

4.2. **Comparison of convergence speed.** The Figure 5 shows the convergence speeds of conditional adversarial domain adaption method(CDAN), Resnet-50, domain-adversarial neural network(DANN) and our proposed method for A to W task. The proposed method convergences at 3999 iterations, whereas CDAN begins to convergence at 8499 iterations, and convergences at 40499 iterations. In this case, the proposed method has the fastest convergence, followed by CDAN method.

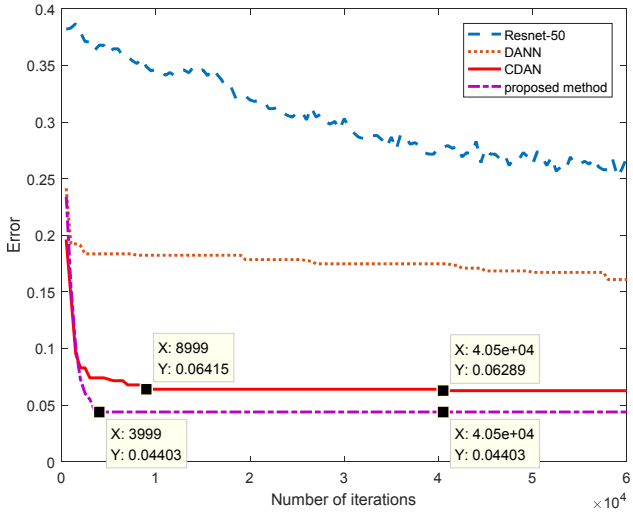

FIGURE 5. Convergence speeds of four methods

4.3. **Comparison of distribution discrepancy.** In this sub-section, we test the distribution discrepancy of our proposed method, domain-adversarial neural network method(DANN), conditional adversarial domain adaption(CDAN) method and Resnet-50 for tasks A to W

and W to A. The A-distance is adopt to measure the distribution discrepancies of these methods. A-distance is defined as:

$$dist_A = 2\,(1 - 2\varepsilon) \tag{26}$$

where ε expresses classifier error. The smaller A-distance is, the more similar the distributions are. In the task A to W, the A-distance results for our proposed method, domain-adversarial neural network method, conditional adversarial domain adaption method and Resnet-50 method are 0.66, 1.44, 1.22 and 1.88, respectively; for task W to A they are 1.3, 1.2, 0.66 and 0.33. In both these two tasks the A-distance of our proposed method is the smallest. Therefore, this show that our proposed method has the best performance on feature extraction (see Figure 6).

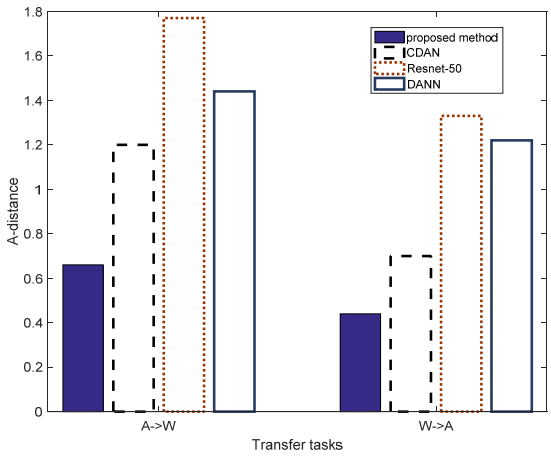

FIGURE 6. Distribution discrepancy results for the conditional adversarial domain adaption method, Resnet-50, domain-adversarial neural network and proposed method

5. **Conclusions.** In this paper, we design a self-attention layer and add it between the fourth layer and fifth layer of Resnet-50, which effectively extracts feature relationships between the current pixel and all other pixels. We also use a least-squares function to deduce new loss and modify the output layer to reduce the generation of gradient disappearance and improve accuracy. Compared with conditional adversarial domain adaptation method, the average accuracies of our proposed method are increased by 2.8%, 1.3% and 6% On the Office-31 dataset, Image CLEF-DA dataset and Office-Home, respectively. For all tasks of the three datasets, the proposed method still has higher accuracy than Resnet-50, deep adaptation network, domain-adversarial neural network, joint adaptation network and conditional adversarial domain adaptation methods. Compared with conditional adversarial domain adaptation method, the proposed method convergences speed is improved by about ten times. Besides, the proposed method also has smaller the A-distance than other methods, which shows that the proposed method has better performance on feature extraction. On the whole, the proposed method has better performance in accuracy and convergence speed than others.

The proposed method is only suitable for the target domain that has the same classifications with source domain. In our feature work, we will consider how to design network to make the method be suitable for target domain that has different classifications with source domain.

Acknowledgment. This work was supported by the National Natural Science Foundation of China (61271115), Research Foundation of Education Bureau of Jilin Province (JJKH20210095KJ), Science and technology development plan innovation project of Jilin City (20190104124).

REFERENCES

[1] K. K. Tseng, J. R. Li, C. M. Chen, M. M. Hassan, A fast instance segmentation with one-stage multi-task deep neural network for autonomous driving. *Computers & Electrical Engineering*, vol. 93, no. 12, 107194, 2021.

[2] R. J. Liu, Y. Q. Shi, C. J. Ji, M. Jia, K. Wang, F. J. Li, C. M. Chen, M. M. Hassan, J. Y. Long, N. Kumar Interpreting Adversarial Examples and Robustness for Deep Learning-Based Auto-Driving Systems, *IEEE Transactions on Intelligent Transportation Systems*, 2021, https://doi.org/10.1109/TITS.2021.3108520.

[3] E. K. Wang, X. Zhang, T. Y. Wu, C. M. Chen, Multilayer Dense Attention Model for Image Caption, *IEEE Access*, vol. 7, pp. 66358 - 66368, 2019.

[4] Y. J. Wang, P. P. Cao, X. S. Wang, X. Y. Yan, Research on Insulator Self Explosion Detection Method Based on Deep Learning, *Journal of Northeast Electric Power University*, vol. 40, no. 3, pp. 33-40, 2020.

[5] Z. X. Zhong, X. B. Ma, A Wei, A Lightweight Solar Irradiance Prediction Model Based on Ground-based Cloud Images and Meteorological Data, *Journal of Northeast Electric Power University*, vol. 41, no. 1, pp. 24-30, 2021.

[6] X. M. Wang, L. Li, W. R. Ye, M. S. Long, Transferable Attention for Domain Adaptation, *AAAI Conference on Artificial Intelligence*, pp. 5345-5352, 2019.

[7] Y. Zhang, H. Tang, K. Jia, M. Tang, Domain-symmetric networks for adversarial domain adaptation, *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5026-5035, 2019.

[8] F. Zhuang, Z. Qi, K. Duan, Y. Zhu, A Comprehensive Survey on Transfer Learning, *AAAI Conference on Artificial Intelligence*, pp. 43-76, 2021.

[9] K. You, M. Long, Z. Cao, J. Wang, M. I. Jordan, Universal Domain Adaptation, *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2715–2724, 2019.

[10] M. Long, Y. Cao, J. Wang, M. I. Jordan, Learning transferable features with deep adaptation networks. *International Conference on International Conference on Machine Learning*, pp. 97-105, 2015.

[11] Y. Zhang, H. Tang, K. Jia, M. Tan, Domain-Symmetric Networks for Adversarial Domain Adaptation, *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5026-5035, 2019.

[12] G. Kang, L. Jiang, Y. Yang, A. G. Hauptmann, Domain-Symmetric Networks for Adversarial Domain Adaptation, *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4888–4897, 2019.

[13] M. Long, H. Zhu, J. Wang, M. I. Jordan, Deep transfer learning with joint adaptation networks, *International Conference on Machine Learning*, pp. 2208-2217, 2017.

[14] J. Wang, Y. Chen, S. Hao, W. Feng, Z. Shen, Balanced Distribution Adaptation for Transfer Learning, *IEEE International Conference on Data Mining (ICDM)*, pp. 1129-1134, 2017.

[15] Y. Li, N. Wang, J. Shi, X. Hou, J. Liu, Adaptive Batch Normalization for practical domain adaptation, *Pattern Recognition*, pp. 109-117, 2018.

[16] F.M. Carlucci, L. Porzi, B. Caputo, E. Ricci, S. R. Bul, AutoDIAL: Automatic DomaIn Alignment Layers, *IEEE International Conference on Computer Vision (ICCV)*, pp. 5077-5085, 2017.

[17] Y. Ganin, V. Lempitsky, Unsupervised Domain Adaptation by Backpropagation, *International Conference on Machine Learning*, pp. 1180-1189, 2017.

[18] E. Tzeng, J. Hoffman, K. Saenko, T. Darrell, Adversarial Discriminative Domain Adaptation, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2962-2971, 2017.

[19] M. Long, Z. Cao, J. Wang, M. I. Jordan, Conditional adversarial domain adaptation. *The 32nd International Conference on Neural Information Processing Systems*, pp. 1647-1657, 2018.

[20] Z. Cao, M. Long, J. Wang, M. I Jordan, Conditional adversarial domain adaptation, *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2724-2732, 2018.

[21] J. Shen, Y. Qu, W. Zhang, Y. Yu, Wasserstein distance guided representation learning for domain adaptation. *AAAI Conference on Artificial Intelligence*, pp. 4058-4065, 2018.

[22] J. Hoffman, E. Tzeng, T. Park, J. Y. Zhu, CyCADA: Cycle-Consistent Adversarial Domain Adaptation, *The 35th International Conference on Machine Learning*, pp. 1989-1998, 2018.

[23] M.Y. Liu, T. Breuel, J. Kautz, Unsupervised image-to-image translation networks, *The 31st International Conference on Neural Information Processing Systems*, pp. 700-708, 2017.

[24] X. Mao, Q. Li, H. Xie, R. Lau, Z. Wang, S. P. Smolley, Least Squares Generative Adversarial Networks. *IEEE International Conference on Computer Vision (ICCV)*, pp. 2813-2821, 2017.

[25] Z. Pei, Z. Cao, M. Long, J. Wang, Multi-adversarial domain adaptation, *AAAI Conference on Artificial Intelligence*, pp. 3934-3941, 2018.